Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Ciencias de la Computación
Programa de Doctorado:
Ingeniería del Software e Inteligencia Artificial

TESIS DOCTORAL

AVANCES EN SISTEMAS SOFTWARE PARA ROBÓTICA CON ATRIBUTOS DE CALIDAD DE SERVICIO ADAPTATIVOS

AUTOR: Juan Adrián Romero Garcés
Ingeniero en Informática

DIRECTOR: Jesús Martínez Cruz
Ingeniero de Telecomunicación
Dr. por la Universidad de Málaga

2017

AUTOR: Juan Adrián Romero Garcés

http://orcid.org/0000-0002-6570-5859

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Índice general

Siglas

ACE Adaptive Communications Environment. 5, 15, 17, 19

DDS Data Distribution Service for real-time systems. 5, 15–17, 19

DSLs Domain Specific Languages. 6

OMG Object Management Group. 15, 17

QoS Quality of Service. 5, 6, 14–17, 19, 20

RMI Remote Method Invocation. 5, 17

Parte I

Departamento de Lenguajes y Ciencias de la Computación
E. T. S. I. Informática
Universidad de Málaga

THESIS SUMMARY

ADVANCES IN SOFTWARE SYSTEMS FOR ROBOTICS WITH ADAPTIVE QUALITY OF SERVICE ATTRIBUTES

AUTHOR: Juan Adrián Romero Garcés
SUPERVISOR: Jesús Martínez Cruz

Thesis summary

1. Objectives and contributions

In this Thesis we present Nerve [Martínez et al., 2012] [Martínez et al., 2011], a middleware for distributed systems and robotics based on the multiplatform *Adaptive Communications Environment (ACE)* toolkit [Schmidt and Huston, 2002] [Schmidt and Huston, 2003] and the *Data Distribution Service for real-time systems (DDS)* [Object Management Group, 2015]. Nerve guarantees the scalability and *Quality of Service (QoS)* needed by real-time systems and it provides users with a self-adapting infraestructure that includes: i) a methodology for monitoring Nerve services through an automatic instrumentation mechanism using the LLVM/Clang compiler infraestructure [Lattner and Adve, 2004] [The clang project, 2016] ; and ii) the adaptation of *QoS* Nerve services to obtain the best performance at run-time. Nerve will also offer users with a *Remote Method Invocation (RMI)* implementation based on *DDS* that will be used to integrate Nerve with different existing frameworks for robotics.

The benefits of Nerve will be proved through its use in real robotics applications: (i) in a robot learning by imitation control architecture and in a visual attention mechanism [Martínez et al., 2012]; (ii) in an audio-visual perception system for a human robotic head [Viciana-Abad et al., 2014]; and (iii) in the cognitive system used by a real social robot [Ingles-Romero et al., 2017].

The main contributions of this Thesis are the following:

1. The development of a middleware for robotics called Nerve that guarantees stringent *QoS* requirements such as real-time and high-performance.

2. The development and integration in Nerve of a *RMI* implementation based on *DDS* [Martínez et al., 2010a].

3. The development of a methodology for monitoring Nerve services. It also provides users with an instrumentation mechanism using the Clang/LLVM compiler infraestructure to help them using this methodology.

4. The development of a self-adaptive infraestructure to adapt *QoS* of Nerve services to obtain the best performance at run-time.

5. The integration in Nerve of a model-driven process for dealing with adaptive *QoS* policies [Ingles-Romero et al., 2017]. This process revolves around the concept of communication template, defined as a prescribed set of *QoS* policies for a particular communication use case.

6. The integration of Nerve with different existing frameworks for robotics: RoboComp [Manso et al., 2010] and SmartSoft [Schlegel and Wörz, 1999]. In the case of RoboComp, and in order to improve its development process, a model-based software development tool was built along with a set of *Domain Specific Languages (DSLs)* to make robocomp middleware-independent [Romero-Garcés et al., 2013b] [Gutierrez Giraldo et al., 2013].

2. Publications

This section lists the publications of the author related with this Thesis:

- Martinez, J., Romero-Garces, A., Rubio, J. P. B., Robles, R. M., and Rubio, A. B. (2012). A dds-based middleware for quality-of-service and high-performance networked robotics. *Concurrency and Computation: Practice and Experience*, 24(16):1940–1952. DOI: 10.1002/cpe.2816, ©2011 John Wiley & Sons Ltd.

- Viciana-Abad, R., Marfil, R., Perez-Lorenzo, J. M., Bandera, J. P., Romero-Garces, A., and Reche-Lopez, P. (2014). Audio-visual perception system for a humanoid robotic head. *Sensors*, 14(6):9522–9545. DOI:10.3390/s140609522

- Martínez, J., Romero-Garcés, A., Vázquez-Martín, R., and Bandera, A. (2010). Recipes for designing high-performance and robust software for robots.In *Robotics Automation and Mechatronics (RAM), 2010 IEEE Conference on*, pages 250–255. IEEE. DOI: 10.1109/RAMECH.2010.5513183. ©2010 IEEE

- Martínez, J., Romero-Garcés, A., Manso, L., and Bustos, P. (2010). Improving a robotics framework with real-time and high-performance features. In *Simulation, Modeling, and Programming for Autonomous Robots*, volume 6472 of *Lecture Notes in Computer Science*, pages 263–274. Springer. DOI: 10.1007/978-3-642-17319-6_26

- Ingles-Romero, J., Romero-Garces, A., Vicente-Chicote, C., and Martinez, J. (2017). A model-driven approach to enable adaptive qos in dds-based middleware. *IEEE Transactions on Emerging Topics in Computational Intelligence*. DOI:10.1109/TETCI.2017.2669187. ©IEEE 2017

- Martinez, J., Romero-Garcés, A., Rubio, J. P. B., and Rubio, A. B. (2011). Nerve: a lightweight middleware for quality-of-service networked robotics. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 655–660. IEEE. DOI: 10.1109/ITNG.2011.116. ©2011 IEEE
- Romero-Garcés, A., Inglés-Romero, J. F., and Martínez, Jesús, C. V.-C. (2013). Self-adaptive quality-of-service in distributed middleware for robotics. In *2nd International Workshop on Recognition and Action for Scene Understanding (REACTS 2013), York (UK)*
- Romero-Garcés, A., Manso, L., Gutierez, M. A., Cintas, R., and Bustos, P. (2013). Improving the lifecycle of robotics components using domain-specific languages. *International Workshop on Domain-Specific Languages and models for ROBotic systems (DSLRob 2011)*
- Gutierrez Giraldo, M. A., Romero-Garces, A., and Bustos, P. (2013). Progress in robocomp. *Journal of Physical Agents*, 7(1):38–47.
  DOI: https://doi.org/10.14198/JoPha.2013.7.1.06

Parte II

Departamento de Lenguajes y Ciencias de la Computación
E. T. S. I. Informática
Universidad de Málaga

RESUMEN DE LA TESIS DOCTORAL

AVANCES EN SISTEMAS SOFTWARE PARA ROBÓTICA CON ATRIBUTOS DE CALIDAD DE SERVICIO ADAPTATIVOS

AUTOR: Juan Adrián Romero Garcés
Ingeniero en Informática

DIRECTOR:

Jesús Martínez Cruz
Ingeniero de Telecomunicación
Dr. por la Universidad de Málaga

1 Introducción

1. Motivación

A diferencia de los robots teleoperados [Vertut, 2013] cuyo objetivo, por ejemplo, puede ser reemplazar a un ser humano en situaciones que entrañen riesgo para éste (como en situaciones de búsqueda y rescate [Murphy, 2004] [Casper and Murphy, 2003]), o de los robots industriales [Nof, 1999] [Craig, 1986], que están destinados a realizar tareas bien definidas y repetitivas en procesos de fabricación industriales, "los robots sociales son robots que trabajan en entornos sociales reales, y que son capaces de percibir, interactuar con y aprender de otros individuos, que suelen ser personas pero pueden ser, también, otros robots sociales" [Bandera, 2010]. Estos robots deben ser capaces de interactuar de forma natural con personas, siendo capaces de reconocerlas a través de algoritmos de visión y/o audio, mantener una conversación, moverse de forma autónoma o manipular objetos. Además, deben ser capaces de desenvolverse en entornos dinámicos y, muchas veces, desconocidos [Francisco Inglés-Romero et al., 2012]. Por ello, están dotados de multitud de sensores y sistemas software complejos que se ejecutan de forma distribuida y que deberían tener un comportamiento predecible y determinista, entrando dentro de la categoría de sistemas empotrados de tiempo real [Brugali and Prassler, 2009].

El software de estos robots suele estar compuesto por multitud de módulos que se encargan de ciertos comportamientos específicos (mecanismo atencional, planificación de tareas, navegación, módulos de aprendizaje, etc.). En la implementación de estos sistemas robóticos se utilizan frameworks que facilitan el desarrollo del software para robots. El enfoque que se ha seguido en los últimos años ha sido mejorar el ciclo de desarrollo software utilizando el desarrollo distribuido basado en componentes, el cual se centra en la reusabilidad y en la evolución del software [Brugali and Prassler, 2009]. De esta manera, el software de un robot está compuesto por componentes distribuidos que se comunican entre sí utilizando un middleware de comunicaciones que permite abstraer al desarrollador de los detalles de bajo nivel, utilizando soluciones ya probadas y permitiendo comunicar aplicaciones en sistemas heterogéneos. Este enfoque ha permitido, entre otras cosas,

afrontar un problema común en cualquier grupo de investigación y equipo de desarrollo: la reutilización y mantenimiento de los módulos software. No obstante, la comunidad robótica es consciente de que existen una serie de requisitos (tiempo real, *QoS*, uso eficiente de recursos, abstracciones heterogéneas, robustez, tolerancia a fallos, mecanismos de configuración y despliegue atomáticos, etc.) que debería satisfacer cualquier software para robótica, tal y como se detalla en los trabajos [Mohamed et al., 2008], [Mohamed et al., 2009] y [Lotz, 2010].

Desafortunadamente, los frameworks y middleware para robótica actuales suelen ofrecer sólo un subconjunto de los requisitos enumerados anteriormente. Es más, su uso requiere un proceso de aprendizaje para poder utilizarlos y satisfacer requisitos de alto rendimiento (como la predecibilidad, eficiencia, escalabilidad, confiabilidad y seguridad) y otras calidades de servicio (*QoS*) como el throughput, jitter, retrasos, probabilidad de pérdidas de datos, tiempo de validez de los mismos, etc. [Martínez et al., 2012] [Martínez et al., 2010b].

El análisis y la verificación de esas propiedades no funcionales debe incluir formas de monitorizar el sistema. Normalmente, las herramientas de monitorización para robótica sólo obtienen información sobre el estado de los componentes y la acción a realizar en caso de que no se cumpla algún requisito es parar el software, lo que suele implicar parar el robot (obviamente este escenario no es la mejor opción para los robots sociales). Por lo tanto, un sistema de control de un robot debería de ser tolerante a fallos y ser capaz de auto-adaptarse [Romero-Garcés et al., 2013a]. Un ejemplo de adaptación podría ser, tal y como se hace en [Francisco Inglés-Romero et al., 2012], optimizar el consumo de energía dependiendo del nivel de la batería. En el caso de que el nivel de batería fuera bajo, el robot podría desplazarse a una velocidad inferior, disminuyendo el consumo. Otro ejemplo podría ser la activación/desactivación de determinados componentes software dependiendo tanto del estado interno del propio robot como del contexto donde se esté ejecutando. En este caso, en un robot equipado con un componente que obtenga imágenes a través de una cámara, se podría desactivar este componente en escenarios donde el nivel de luz fuera insuficiente, y volverlo a activar en caso de que se detectara un nivel adecuado.

Los diseñadores de software para robots suelen mitigar esos problemas utilizando herramientas de simulación (como Stage [Gerkey et al., 2003], Gazebo3D [Koenig and Howard, 2004] o RoboComp InnerModel Simulator [Gutierrez Giraldo et al., 2013]) que permiten probar sus algoritmos en entornos similares a donde se utilizará el sistema final. No obstante, a pesar de que estas herramientas son útiles para identificar comportamientos anómalos del robot, no pueden garantizar el correcto funcionamiento del sistema en cualquier situación real futura.

Hasta el momento, no suele ser común que los frameworks para robótica ofrezcan la posibilidad de adaptar el software en tiempo de ejecución, un requisito esencial para un robot social. Sin embargo, se han hecho grandes avances en los últimos años. La

auto-configuración suele centrarse en robots fabricados con una estructura hardware modular que permite modificar su forma para adaptarse a su entorno y poder realizar diferentes acciones [Yim et al., 2007], o en la adaptación de componentes software dependiendo del estado del robot y de las condiciones del entorno (ver [Lotz, 2010], [Georgas and Taylor, 2008] y [Edwards et al., 2009]).

No obstante, la auto-adapación del sistema debería abarcar no sólo a la lógica de los componentes, sino también al middleware de comunicaciones que se esté utilizando, ya que puede ser necesario ajustar algunas de sus calidades de servicio (*QoS*) para mejorar el rendimiento. El middleware de comunicaciones juega un papel fundamental en el comportamiento del sistema ya que, normalmente, los recursos del sistema (consumo de CPU, memoria, uso de la red, etc.) pueden variar en tiempo de ejecución [Ingles-Romero et al., 2017]. La adaptación a nivel de middleware de comunicaciones es esencial para garantizar el uso eficiente de los recursos y el rendimiento esperado.

2. Objetivos

En esta Tesis se presenta Nerve [Martínez et al., 2012] [Martínez et al., 2011], un middleware para sistemas distribuidos y robótica basado en la librería de comunicaciones *ACE* [Schmidt and Huston, 2002] [Schmidt and Huston, 2003] y en el middleware de comunicaciones *DDS* [Object Management Group, 2015]. *DDS* es el primer middleware estandarizado por el *Object Management Group (OMG)* que implementa el modelo publicación/suscripción para sistemas empotrados de tiempo real. El estándar *DDS* define veintidós *QoS* para especificar el límite de recursos, el tamaño de las colas de datos, la fiabilidad, la viveza de las entidades participantes, el tiempo de vida de los datos, o la prioridad en la entrega de datos, por nombrar algunas. Esto ofrece muchas posibilidades a los usuarios a la hora de configurar las *QoS* de sus aplicaciones. No obstante, la selección adecuada de dichas *QoS* para obtener el mejor rendimiento posible no es una tarea sencilla. La solución típica consiste en utilizar un conjunto estático de *QoS* durante la ejecución del sistema. Sin embargo, aunque ese conjunto estático de *QoS* puede ser la solución adecuada en determinadas circunstancias, puede no ser la óptima si las condiciones cambian en tiempo de ejecución [Ingles-Romero et al., 2017]. Además, modificar las *QoS* en tiempo de ejecución puede llevar a incompatibilidades, ya que las *QoS* de las diferentes entidades en *DDS* deben ser consistentes entre sí [Object Management Group, 2015].

Se ha implementado una solución con la cual es posible adaptar de forma automática las políticas de *QoS* de Nerve para obtener el mejor rendimiento posible en tiempo de ejecución. Para ello, Nerve ofrece una infraestructura de monitorización y adaptación. La monitorización permite tener acceso a métricas del sistema (como el consumo de CPU, memoria, uso de la red, etc.) y a variables de interés en el código de los usuarios (como las *QoS* de las diferentes entidades utilizadas o las variables utilizadas dentro de sus

algoritmos). También se ha implementado un mecanismo de instrumentación automática de código utilizando la infraestructura de compiladores LLVM [Lattner and Adve, 2004] y en concreto, el compilador Clang [The clang project, 2016], que permite modificar el código del usuario para monitorizar dichas variables. Así, el usuario podrá especificar en un fichero de configuración las variables que desea monitorizar dentro de su programa y, llevando a cabo transformaciones en el código intermedio generado durante el proceso de compilación, se añadirán de forma automática los objetos y métodos necesarios para llevar a cabo la monitorización. Esta característica evita que el usuario tenga que modificar manualmente el código para monitorizar variables y métricas del sistema.

Con respecto a la infraestructura de adaptación, Nerve permite modificar las *QoS* de las entidades *DDS* utilizadas en el sistema según la información de monitorización y los requisitos de la aplicación. Para validar esta propuesta se ha utilizado un enfoque dirigido por modelos junto con una serie de lenguajes de dominio específico implementados por investigadores de la Universidad de Cartagena y la Universidad de Extremadura, con los que se proporciona a los diseñadores los medios necesarios para especificar los límites de variabilidad de ciertas *QoS* y prevenir la aparición de configuraciones de *QoS* incompatibles en tiempo de ejecución, tal y como se describe en [Ingles-Romero et al., 2017] (ver también la primera versión implementada en [Romero-Garcés et al., 2013a]). Este enfoque puede ser utilizado por cualquier middleware basado en *DDS* y ha sido integrado y validado en Nerve.

Además, para ofrecer a los usuarios de Nerve una mayor flexibilidad a la hora de decidir el modelo de comunicaciones a utilizar, también se ha implementado un mecanismo de comunicaciones basado en petición/respuesta utilizando *DDS* [Martínez et al., 2010a] y que ha sido integrado en Nerve. De esta forma se ha conseguido que Nerve proporcione tanto el modelo de comunicaciones publicación/suscriptor como el modelo cliente/servidor. Este nuevo modelo de comunicaciones se ha utilizado para integrar Nerve en dos frameworks para robótica: SmartSoft [Schlegel and Wörz, 1999] y RoboComp [Manso et al., 2010].

Finalmente, los beneficios del uso de Nerve se han reflejado en su aplicación en sistemas robóticos reales. En concreto, en dos sistemas cognitivos (uno basado en aprendizaje y visión [Martínez et al., 2012], y otro en percepción audio-visual [Viciana-Abad et al., 2014]) y en un subconjunto del sistema cognitivo utilizado por un robot social real [Romero-Garcés et al., 2015] [Ingles-Romero et al., 2017].

3. Contribuciones

El núcleo central de esta Tesis es el desarrollo del middleware Nerve, que permite construir servicios robóticos con *QoS*. Este ha sido ampliado con una serie de

funcionalidades (incluyendo *RMI*) para permitir construir servicios auto-adaptativos, y se ha integrado en dos frameworks para robótica existentes.

Las contribuciones de esta Tesis son las siguientes:

1. El desarrollo de Nerve, un middleware para robótica con requisitos de tiempo real utilizando el estándar de distribución de datos del *OMG* (*DDS*) y la librería de comunicaciones *ACE*.

2. Descripción, implementación e integración en Nerve de un mecanismo de comunicaciones *RMI* basado en *DDS*.

3. Desarrollo de un mecanismo de monitorización para el middleware Nerve que permita monitorizar variables del sistema y de interés por parte del usuario. Se proporcionará un mecanismo de instrumentación automática de código utilizando la infraestructura de compiladores LLVM para facilitar su uso por parte del usuario.

4. Implementación de una infraestructura de adaptación que permitirá que cualquier sistema basado en Nerve adapte sus *QoS* en tiempo de ejecución con el objetivo de conseguir el mejor rendimiento con los recursos disponibles.

5. Integración en Nerve de una metodología de alto nivel (basada en modelos) ya existente para el modelado de *QoS* adaptativas. Para ello, se utilizará el concepto de plantilla de comunicación como mecanismo de abstracción de alto nivel para especificar políticas de *QoS* en escenarios distribuidos. Las políticas de *QoS* adaptativas se definirán a través de modelos de adaptación asociadas a dichas plantillas.

6. Integración de Nerve en dos frameworks para robótica: RoboComp y SmartSoft. En el caso del framework RoboComp, y con el objetivo de mejorar el proceso de desarrollo de sus componentes y de facilitar la integración de otros middleware de comunicaciones en dicho framework, se ha implementado una infraestructura asentada en el desarrollo de software basado en modelos que proporciona: 1) una serie de lenguajes de dominio específico y la infraestructura necesaria para generar componentes RoboComp desde esos lenguajes, tal y como se describe en [Romero-Garcés et al., 2013b]; y 2) una extensión de dichos lenguajes para hacer al framework RoboComp independiente del middleware de comunicaciones [Gutierrez Giraldo et al., 2013].

2

Conclusiones y trabajo futuro

En esta Tesis se ha implementado Nerve, un middleware para sistemas distribuidos y robótica con atributos de calidad de servicio basado en *ACE* y en el estándar *DDS*. Nerve permite construir servicios robóticos con requisitos de *QoS* tales como tiempo real y alto rendimiento, y ofrece un mecanismo para auto-adaptar las *QoS* en tiempo de ejecución para poder obtener el mejor rendimiento posible. Además, se ha utilizado en la implementación de tres subsistemas claves en cualquier sistema cognitivo de un robot social:

- El mecanismo de atención visual, que se utiliza para obtener información relevante a partir de las imágenes percibidas.
- Una arquirectura de control basada en aprendizaje por imitación, que permite a un robot social aprender de personas utilizando los mismos mecanismos de comunicación que serían utilizados para enseñar a otros seres humanos.
- Un mecanismo de percepción audio-visual capaz de calcular la posición de una persona a partir de las imágenes y el audio obtenidos, y de mover una cabeza robótica hacia ella.

Al integrar *DDS*, Nerve permite configurar las aplicaciones con amplio conjunto de *QoS*. Sin embargo, la selección y configuración de las mismas no es una tarea sencilla para los desarrolladores inexpertos. Nerve simplifica este proceso permitiendo la configuración de *QoS* a través de un conjunto preestablecido de valores y es capaz de adaptar dinámicamente las políticas de *QoS* para proporcionar el mejor rendimiento posible con los recursos disponibles. Para ello, se ha implementado la infraestructura necesaria para llevar a cabo tanto la monitorización (incluyendo un mecanismo de instrumentación automática de código utilizando LLVM y Clang) como la adaptación de *QoS* en tiempo de ejecución. En este sentido, en esta Tesis se han afrontado los siguientes desafíos:

- La complejidad de seleccionar y configurar las políticas de *QoS* apropiadas.

- La adaptación dinámica de políticas de *QoS* para conseguir el mejor rendimiento posible.

Para afrontar estos problemas se ha utilizado el concepto de plantilla de comunicación como mecanismo de abstracción para especificar *QoS* en escenarios distribuidos junto al uso de modelos de adaptación para definir políticas de *QoS* adaptativas. Este proceso ha sido integrado en Nerve y utiliza un enfoque basado en el desarrollo de software dirigido por modelos en el cual dos lenguajes de modelado se utilizan para describir la lógica de adaptación.

Como línea de trabajo futuro se podría plantear la posibilidad de generar servicios basados en Nerve a partir de los modelos visuales de SmartSoft y extender dichos modelos para poder especificar y verificar las *QoS*. También se podría extender el trabajo presentado en esta Tesis para explorar la adaptación en protocolos de transporte y sus *QoS*, como se hace en [Hoffert and Schmidt, 2009]. Adicionalmente, se podría investigar una adaptación a varios niveles, por ejemplo, para balancear los recursos y el rendimiento a nivel del middleware de comunicaciones mientras el sistema optimiza simultáneamente otras propiedades no funcionales a nivel de aplicación.

Finalmente, otra línea de trabajo futuro podría ser extender el trabajo presentado en esta Tesis con herramientas de verificación en tiempo de ejecución que analicen requisitos de *QoS* (como hace [Havelund, 2008]), tanto en el middleware de comunicaciones como en el propio framework para robótica.

3

Listado de publicaciones

Se enumera a continuación la producción científica relacionada con esta Tesis:

- Martínez, J., Romero-Garces, A., Rubio, J. P. B., Robles, R. M., and Rubio, A. B. (2012). A dds-based middleware for quality-of-service and high-performance networked robotics. *Concurrency and Computation: Practice and Experience*, 24(16):1940–1952. DOI: 10.1002/cpe.2816, ©2011 John Wiley & Sons Ltd

- Viciana-Abad, R., Marfil, R., Perez-Lorenzo, J. M., Bandera, J. P., Romero-Garces, A., and Reche-Lopez, P. (2014). Audio-visual perception system for a humanoid robotic head. *Sensors*, 14(6):9522–9545. DOI:10.3390/s140609522

- Martínez, J., Romero-Garcés, A., Vázquez-Martín, R., and Bandera, A. (2010b). Recipes for designing high-performance and robust software for robots. In *Robotics Automation and Mechatronics (RAM), 2010 IEEE Conference on*, pages 250–255. IEEE. DOI: 10.1109/RAMECH.2010.5513183. ©2010 IEEE

- Martínez, J., Romero-Garcés, A., Manso, L., and Bustos, P. (2010a). Improving a robotics framework with real-time and high-performance features. In *Simulation, Modeling, and Programming for Autonomous Robots*, volume 6472 of *Lecture Notes in Computer Science*, pages 263–274. Springer. DOI: 10.1007/978-3-642-17319-6_26

- Ingles-Romero, J., Romero-Garces, A., Vicente-Chicote, C., and Martinez, J. (2017). A model-driven approach to enable adaptive qos in dds-based middleware. *IEEE Transactions on Emerging Topics in Computational Intelligence*. DOI:10.1109/TETCI.2017.2669187, ©IEEE 2017

- Martínez, J., Romero-Garcés, A., Rubio, J. P. B., and Rubio, A. B. (2011). Nerve: a lightweight middleware for quality-of-service networked robotics. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 655–660. IEEE. DOI: 10.1109/ITNG.2011.116, ©2011 IEEE

- Romero-Garcés, A., Inglés-Romero, J. F., and Martínez, Jesús, C. V.-C. (2013a). Self-adaptive quality-of-service in distributed middleware for robotics. In *2nd International Workshop on Recognition and Action for Scene Understanding (REACTS 2013), York (UK)*
- Romero-Garcés, A., Manso, L., Gutierez, M. A., Cintas, R., and Bustos, P. (2013b). Improving the lifecycle of robotics components using domain-specific languages. *International Workshop on Domain-Specific Languages and models for ROBotic systems (DSLRob 2011)*
- Gutierrez Giraldo, M. A., Romero-Garces, A., and Bustos, P. (2013). Progress in robocomp. *Journal of Physical Agents*, 7(1):38–47. DOI: https://doi.org/10.14198/JoPha.2013.7.1.06

Bibliografía

[Bandera, 2010] Bandera, J. P. (2010). *Vision-based gesture recognition in a robot learning by imitation framework*. Ph.d. dissertation, Available at http://www.grupoisis.uma.es. Dpto. Tecnología Electrónica, Universidad de Málaga, Spain.

[Brugali and Prassler, 2009] Brugali, D. and Prassler, E. (2009). Software engineering for robotics. *IEEE Robotics and Automation Magazine*, 16(1):9–15.

[Casper and Murphy, 2003] Casper, J. and Murphy, R. R. (2003). Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 33(3):367–385.

[Craig, 1986] Craig, J. (1986). *Introduction to robotics: mechanics and control*. Addison-Wesley, Boston, MA, USA.

[Edwards et al., 2009] Edwards, G., Garcia, J., Tajalli, H., Popescu, D., Medvidovic, N., Sukhatme, G., and Petrus, B. (2009). Architecture-driven self-adaptation and self-management in robotics systems. In *Software Engineering for Adaptive and Self-Managing Systems, 2009. SEAMS'09. ICSE Workshop on*, pages 142–151. IEEE.

[Francisco Inglés-Romero et al., 2012] Francisco Inglés-Romero, J., Lotz, A., Chicote, C. V., and Schlegel, C. (2012). Dealing with run-time variability in service robotics: Towards a dsl for non-functional properties. *International Workshop on Domain-Specific Languages and models for ROBotic systems (DSLRob 2012)*.

[Georgas and Taylor, 2008] Georgas, J. C. and Taylor, R. N. (2008). Policy-based self-adaptive architectures: a feasibility study in the robotics domain. In *Proceedings of the 2008 international workshop on Software engineering for adaptive and self-managing systems*, pages 105–112. ACM.

[Gerkey et al., 2003] Gerkey, B., Vaughan, R. T., and Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th international conference on advanced robotics*, volume 1, pages 317–323.

[Gutierrez Giraldo et al., 2013] Gutierrez Giraldo, M. A., Romero-Garces, A., and Bustos, P. (2013). Progress in robocomp. *Journal of Physical Agents*, 7(1):38–47. DOI: https://doi.org/10.14198/JoPha.2013.7.1.06.

[Havelund, 2008] Havelund, K. (2008). Runtime verification of c programs. In *Testing of Software and Communicating Systems*, pages 7–22. Springer.

[Hoffert and Schmidt, 2009] Hoffert, J. and Schmidt, D. C. (2009). Maintaining qos for publish/subscribe middleware in dynamic environments. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09, pages 28:1–28:2, New York, NY, USA. ACM. DOI: 10.1145/1619258.1619295.

[Ingles-Romero et al., 2017] Ingles-Romero, J., Romero-Garces, A., Vicente-Chicote, C., and Martinez, J. (2017). A model-driven approach to enable adaptive qos in dds-based middleware. *IEEE Transactions on Emerging Topics in Computational Intelligence*. DOI:10.1109/TETCI.2017.2669187, ©IEEE 2017.

[Koenig and Howard, 2004] Koenig, N. and Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2149–2154. IEEE.

[Lattner and Adve, 2004] Lattner, C. and Adve, V. (2004). Llvm: A compilation framework for lifelong program analysis & transformation. In *Code Generation and Optimization, 2004. CGO 2004. International Symposium on*, pages 75–86. IEEE.

[Lotz, 2010] Lotz, A. (2010). Monitoring in Robotic Systems. Master's thesis, Hochshule Ulm - University of Applied Sciences, Ulm.

[Manso et al., 2010] Manso, L., Bachiller, P., Bustos, P., Núñez, P., Cintas, R., and Calderita, L. (2010). Robocomp: a tool-based robotics framework. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 251–262. Springer.

[Martínez et al., 2010a] Martínez, J., Romero-Garcés, A., Manso, L., and Bustos, P. (2010a). Improving a robotics framework with real-time and high-performance features. In *Simulation, Modeling, and Programming for Autonomous Robots*, volume 6472 of *Lecture Notes in Computer Science*, pages 263–274. Springer. DOI: 10.1007/978-3-642-17319-6_26.

[Martínez et al., 2012] Martínez, J., Romero-Garces, A., Rubio, J. P. B., Robles, R. M., and Rubio, A. B. (2012). A dds-based middleware for quality-of-service and high-performance networked robotics. *Concurrency and Computation: Practice and Experience*, 24(16):1940–1952. DOI: 10.1002/cpe.2816, ©2011 John Wiley & Sons Ltd.

[Martínez et al., 2011] Martínez, J., Romero-Garcés, A., Rubio, J. P. B., and Rubio, A. B. (2011). Nerve: a lightweight middleware for quality-of-service networked robotics. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 655–660. IEEE. DOI: 10.1109/ITNG.2011.116, ©2011 IEEE.

[Martínez et al., 2010b] Martínez, J., Romero-Garcés, A., Vázquez-Martín, R., and Bandera, A. (2010b). Recipes for designing high-performance and robust software for robots. In *Robotics Automation and Mechatronics (RAM), 2010 IEEE Conference on*, pages 250–255. IEEE. DOI: 10.1109/RAMECH.2010.5513183. ©2010 IEEE.

[Mohamed et al., 2008] Mohamed, N., Al-Jaroodi, J., and Jawhar, I. (2008). Middleware for robotics: A survey. In *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, pages 736–742. IEEE.

[Mohamed et al., 2009] Mohamed, N., Al-Jaroodi, J., and Jawhar, I. (2009). A review of middleware for networked robots. *International Journal of Computer Science and Network Security*, 9(5):139–148.

[Murphy, 2004] Murphy, R. R. (2004). Human-robot interaction in rescue robotics. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(2):138–153.

[Nof, 1999] Nof, S. Y. (1999). *Handbook of industrial robotics*, volume 1. John Wiley & Sons.

[Object Management Group, 2015] Object Management Group (2015). Data Distribution Service for Real-time Systems (DDS), version 1.4. Available at http://www.omg.org/spec/DDS/.

[Romero-Garcés et al., 2013a] Romero-Garcés, A., Inglés-Romero, J. F., and Martínez, Jesús, C. V.-C. (2013a). Self-adaptive quality-of-service in distributed middleware for robotics. In *2nd International Workshop on Recognition and Action for Scene Understanding (REACTS 2013), York (UK)*.

[Romero-Garcés et al., 2013b] Romero-Garcés, A., Manso, L., Gutierez, M. A., Cintas, R., and Bustos, P. (2013b). Improving the lifecycle of robotics components using domain-specific languages. *International Workshop on Domain-Specific Languages and models for ROBotic systems (DSLRob 2011)*.

[Romero-Garcés et al., 2015] Romero-Garcés, A. n., Calderita, L. V., Martínez-Gómez, J., Bandera, J. P., Marfil, R., Manso, L. J., Bandera, A., and Bustos, P. (2015). Testing a fully autonomous robotic salesman in real scenarios. In *Autonomous Robot Systems and Competitions (ICARSC), 2015 IEEE International Conference on*, pages 124–130. IEEE.

[Schlegel and Wörz, 1999] Schlegel, C. and Wörz, R. (1999). The software framework smartsoft for implementing sensorimotor systems. In *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, volume 3, pages 1610–1616. IEEE.

[Schmidt and Huston, 2002] Schmidt, D. C. and Huston, S. (2002). *C++ Network Programming Volume 1: Mastering Complexity with ACE and Patterns*. Addison-Wesley.

[Schmidt and Huston, 2003] Schmidt, D. C. and Huston, S. (2003). *C++ Network Programming Volume 2: Systematic Reuse with ACE and Frameworks*. Addison-Wesley.

[The clang project, 2016] The clang project (2016). clang: a C language family frontend for LLVM. Available at http://clang.llvm.org/.

[Vertut, 2013] Vertut, J. (2013). *Teleoperation and robotics: applications and technology*, volume 3. Springer Science & Business Media.

[Viciana-Abad et al., 2014] Viciana-Abad, R., Marfil, R., Perez-Lorenzo, J. M., Bandera, J. P., Romero-Garces, A., and Reche-Lopez, P. (2014). Audio-visual perception system for a humanoid robotic head. *Sensors*, 14(6):9522–9545. DOI:10.3390/s140609522.

[Yim et al., 2007] Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., and Chirikjian, G. S. (2007). Modular self-reconfigurable robot systems [grand challenges of robotics]. *Robotics & Automation Magazine, IEEE*, 14(1):43–52.